INSTITUT D'ENSEIGNEMENT SUPÉRIEUR DE RUHENGERI

Scientia et Lux

Accredited by Ministerial Order N°005/2010/MINÉDUC of 16 June 2010

16th March, 2025

TECHNICAL REPORT

Faculty: AFS
Department: Computer Science
class: SWE A year3
Course: Artificial intelligence
Project Name: AI challenge

Group Members:

Names	RegNo
Abari Ilior Aichetou	23/21291
Izere Fidella	23/19439
Munyaneza Emmanuel	23/20865
Chol Adut Gai	23/21358
UMUHUZA Assoumpta	23/21209
Byiringiro Germain	23/21411
Denyse NYIRATUZA	23/23420
Abakar Sidick Baba Abakar	22/19137
GIHOZO Brigitte	23/23271
Uwera Gloriose	23/20525

B.P. 155

Ruhengeri
Rwanda

T : +250 788 90 30 30

F : +250 788 90 30 32

W : www.ines.ac.rw

1. Problem Statement & Justification

Crop diseases are a significant threat to global food security, that causes substantial losses in agricultural productivity and thus, to a country's economy. Diseases like wheat rust, potato blight, and rice blast can devastate entire crops, leading to economic losses and food shortages. Early detection and prevention are critical to mitigating these impacts. Additionnaly, farmers also need a system that doesn't just diagnose diseases but helps them keep their plants healthy by giving real-time advice and recommendations in order to maintain optimal plant health

Example: In 2020, a severe outbreak of wheat stem rust in East Africa led to a 30% reduction in wheat yields, affecting millions of farmers.

Impact: Economic losses, increased food prices, and reduced food availability.

This system aims to resolve this problem and bridge the gap by providing farmers with an AI driven solution. This approach was chosen, for traditional methods of disease detection rely on manual inspection by experts, which is time-consuming, labor-intensive, not really reliable, and often inaccurate. AI, particularly expert system in our case is thus the perfect solution as it does not require high computational cost, is available for use 24/7, and gives good results.

2. Data Collection & Preprocessing

To develop this expert system for plant disease detection and prevention, we gathered and analyzed multiple datasets covering both environmental factors and visual symptoms of plant diseases. Our approach involved structured data acquisition, preprocessing, and enrichment through external research.

i. Data Collection

We sourced our datasets from PlantVillage and Kaggle, two well-known platforms that we found on the internet, they provide high-quality agricultural datasets. The data used in this project can be categorized into two main types:

- Weather and Soil Nutrients Data:

This dataset includes information on how environmental factors such as rainfall, temperature, humidity, and soil nutrient levels (e.g., nitrogen, phosphorus, and potassium) impact plant health. The goal was to extract insights that help in early disease prediction and overall crop health assessment.

- Plant Disease Image Dataset:

A collection of labeled images representing various plant diseases affecting maize, coffee, avocado, and banana crops. The dataset was used to train and validate the system's ability to recognize disease symptoms visually.

ii. Data Exploration and Enrichment

To ensure accuracy and relevance, we complemented our datasets with online research focused on the four selected plant types. We studied disease symptoms, their causes, and best treatment practices from scientific literature, agricultural extension services, and plant pathology reports. This helped refine the system's diagnostic logic and recommendations.

iii. Data Preprocessing

Before integrating the data into the expert system, we applied the following preprocessing steps:

- Cleaning: Removed inconsistencies, missing values, and redundant records in the tabular dataset.

- Feature Selection: Focused on key parameters (e.g., temperature thresholds, soil nutrient levels) that directly influence plant health.

- Image Processing: Applied image resizing, normalization, and augmentation to improve disease classification performance.

-Standardization: Converted measurement units where necessary to maintain consistency across different data sources.

iv. Insights Extraction

Through our data analysis, we identified patterns linking weather conditions and soil health to plant disease occurrences. This allowed us to refine our expert system's decision-making process, ensuring that users receive precise diagnoses and prevention recommendations based on real-world data.

3. AI Model Development

For implementation, we structured the AI model around a rule-based inference engine that relies on predefined symptom-disease relationships. The goal was to create an expert system that could analyze user-input symptoms and provide accurate recommendations for disease diagnosis and plant care.

i. Choice of Framework: ***Streamlit***

We selected Streamlit as our development framework because it simplifies both backend logic and frontend deployment. Streamlit allows us to quickly prototype and implement an intuitive user interface while integrating a responsive decision-making system. Since it runs as a web application, users can easily access it without requiring extensive software installations.

ii. Knowledge Representation and Development Process

- Dataset Utilization:

- We used PlantVillage and Kaggle datasets to study the impact of weather conditions and soil nutrients on plant health.

- The image datasets provided insight into the common diseases affecting cassava and other types of crops, helping us establish clear symptom-disease associations.

- Additional research was conducted to verify the accuracy of symptom-based disease detection and to find more plant diseases that would interest us.

- Rule-Based Inference Engine:

- The system relies on a predefined set of symptoms and their corresponding diseases.

- Users select symptoms through checkboxes, and the system matches them with known disease patterns.

- If a disease is detected, the system provides a diagnosis, recommended treatment, and preventive measures.

- If no match is found, the system prompts the user to input more symptoms or provides general plant health maintenance advice.

iii. System Workflow

1. User selects a plant type → The system dynamically displays relevant symptoms.

2. User checks applicable symptoms→ The inference engine matches symptoms with known diseases.

3. If a disease is detected → The system displays the diagnosis, treatment recommendations, and prevention tips.

4. If no disease is identified → The system asks for additional symptoms or provides general plant care advice.

iv. Deployment and Accessibility

With Streamlit, the expert system is lightweight and easy to deploy. It can be accessed via a web browser, allowing farmers and agricultural experts to use it on-site without complex installations. This makes it a practical tool for real-time crop health monitoring and decision-making.

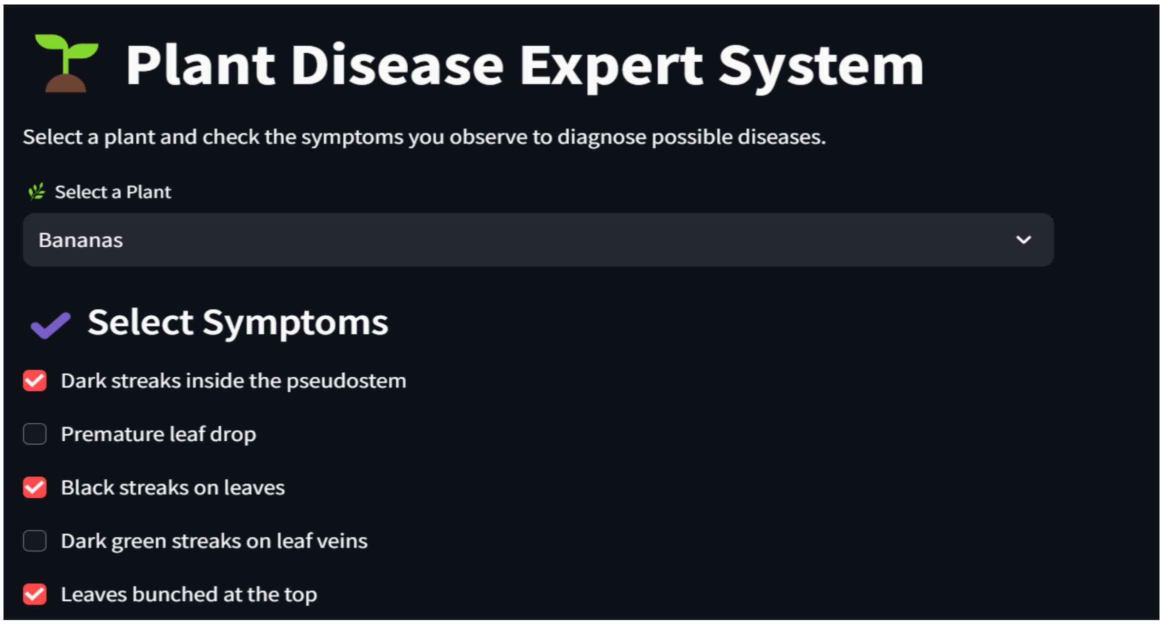
By combining structured knowledge from datasets with a rule-based expert system, we developed a deep, symptom-based plant disease detection tool. This system helps farmers and agricultural professionals make informed decisions about cassava, maize, coffee, avocado, and banana crops, ensuring better disease management and improved plant health.

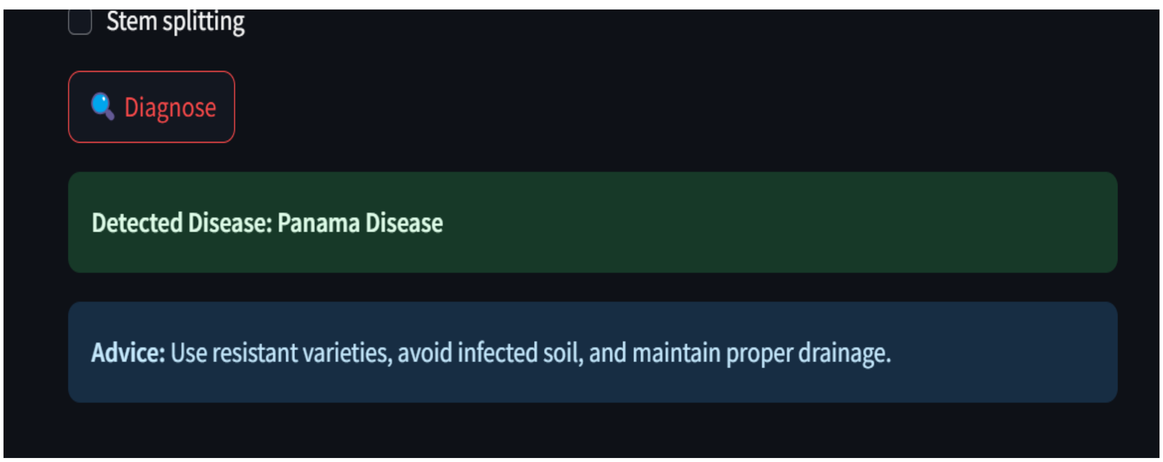
4. Testing & Model Validation

The system went through a series of tests like unit testing, functional testing, integration testing, user acceptance testing, system testing, regression testing and system testing. For extra validation the system was tested twice, the first was to verify the functionality and some changes were made and the second was the perfection of the system.

Unit testing:

During the unit testing, each component was tested individually i.e, the plant selection of banana was tested by adding different symptoms to test if the diagnosis will be made according to the symptoms entered and we made sure that with the different symptoms added the system displays the appropriate diseases to match the symptoms. And the same was done for the rest the plants accordingly.





User acceptance testing:

During the user acceptance testing the system was tested to see if it is user friendly and this test was done by both the development team and the other classmates that are not part of the group who verified that the system is indeed user friendly.

Functional testing:

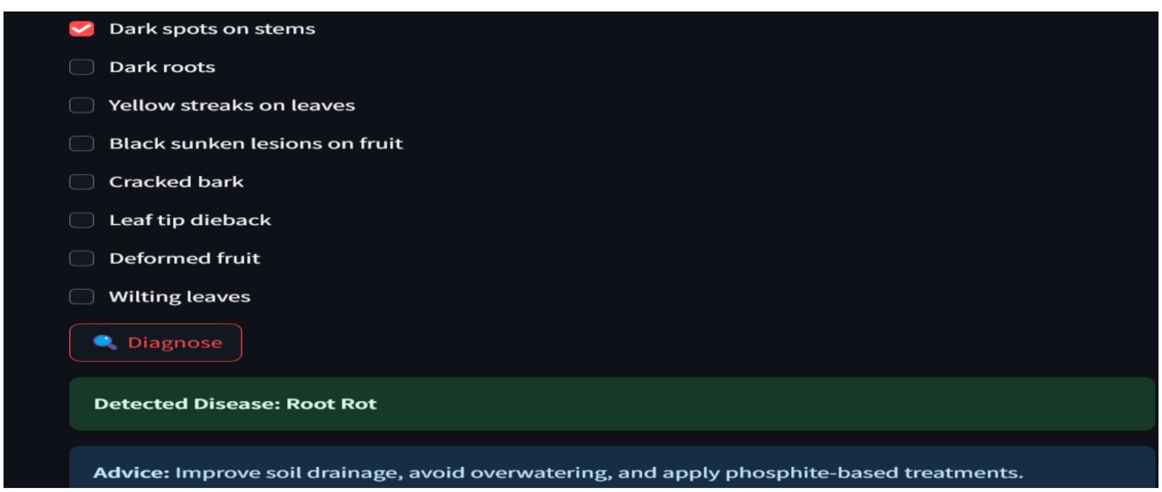
The system was tested to see if it meets the functional requirements that were stated at the requirements gathering stage.

Regression testing:

After the first testing, the system underwent some changes, and the second testing was applied to check whether the changes affected the existing functionality.

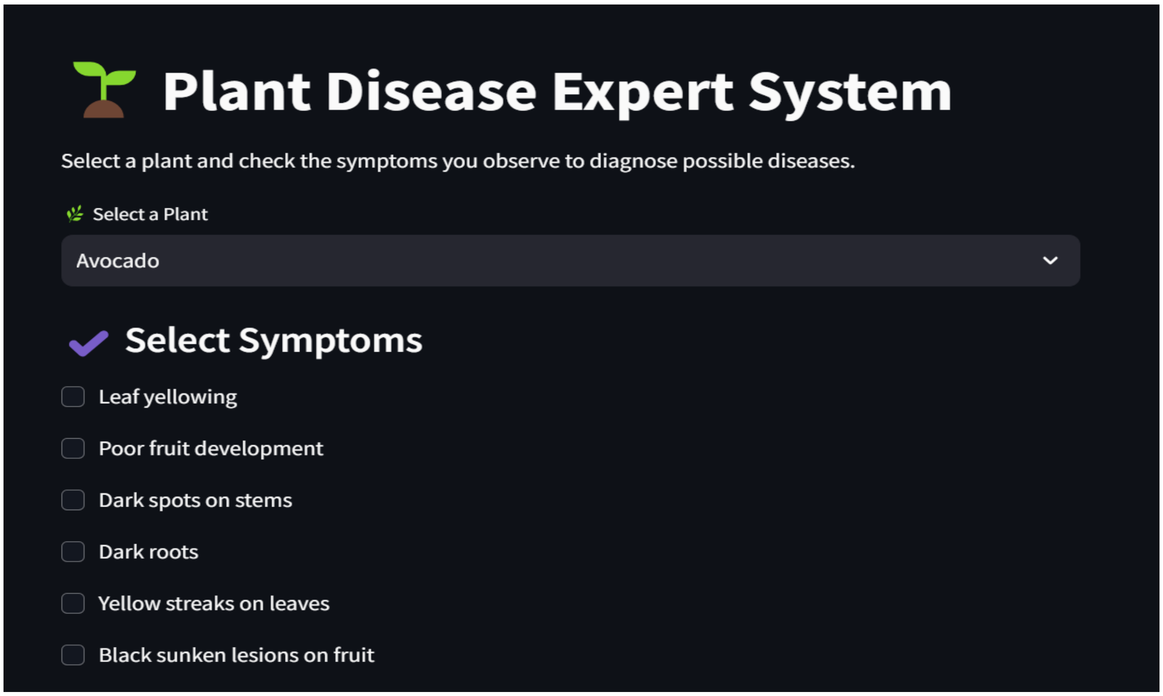
Integration testing:

The system was tested to see if the different components work together e.g the symptoms function and the diseases. We checked if the different symptoms display the appropriate diseases and how the different functions work together to produce the expected results.



System testing:

The system was tested entirely to ensure that it functions as expected. Different diseases were selected with different symptoms every time and the system kept displaying the appropriate diseases. We continued with entering different symptoms for the same plants and displays the appropriate diseases still. The test was done with all the plants that are displayed in the system, and it works as expected.



5. AI System Deployment & User Interaction

To ensure easy access and usability, we deployed the expert system as a web application using Streamlit. This choice allows users to interact with the system through a simple browser interface without needing to install additional software. The system deployed on Streamlit Community Cloud for public access.

i. How Users Can Access the System?

- Web Interface: Users can open the application in any modern web browser.

- Local Execution: The system can be run locally using `streamlit run app.py`.

- Cloud platform: The system is hosted at https://aigroup1implementationassignment3-pvpq34r5akikb9tezwb2nr.streamlit.app/ to provide remote access.

ii. User Interaction Workflow

1. Select a Plant Type

The user starts by choosing a plant (cassava, Maize, Coffee, Avocado, or Banana) from a dropdown menu.

2. Providing Symptoms

- Once a plant is selected, a list of possible symptoms appears as checkboxes.
- The user selects the symptoms observed on their plant.

3. Receiving Diagnosis and Advice

- The system processes the selected symptoms using rule-based inference(IF THEN logic).
- If a disease is identified, the system displays:
  - The disease name
  - Treatment recommendations
  - Preventive measures
- If no disease is found, the system will either:
  - Request more symptoms for a better diagnosis.
  - Provide general plant health maintenance tips if the plant appears healthy.

4. Acting on Recommendations

Users can follow the suggested treatments and return to the system if symptoms persist or worsen.

iii. User-Friendly Features

✅ Interactive UI– The checkboxes make it easy to input symptoms.

✅ Real-time Processing– The system provides instant results without delays.

✅ No Technical Expertise Needed – Farmers and agricultural experts can use the system with minimal training.

✅ Mobile-Friendly – The web-based approach ensures accessibility from smartphones, tablets, and desktops.

This expert system is designed to be simple, fast, and accessible. By leveraging Streamlit's interactive features, users can easily diagnose plant diseases, receive treatment advice, and maintain crop health, all within a few clicks.

6. Challenges Faced & Future Improvements

Challenges:

i. Limited Access to Comprehensive Datasets

We used datasets from PlantVillage and Kaggle, but they primarily contained weather, soil nutrients, and plant disease classifications. Finding datasets that linked disease symptoms to specific diagnoses required additional manual research to fill in gaps.

ii. Defining Accurate Symptom-Disease Relationships

Creating a deep expert system meant ensuring that symptom selections led to the correct disease diagnosis. We had to carefully structure the knowledge base to avoid false positives or missed diagnoses.

iii. Balancing System Depth with Simplicity

We wanted the system to be detailed and comprehensive while keeping it easy to use for farmers. Too many symptoms or complex decision paths could have made the system overwhelming, so we optimized the user interface and logic flow.

iv. Deploying for Accessibility

Streamlit made deployment easier, but making the system accessible beyond local execution still required exploring cloud hosting options. Ensuring that the system runs smoothly across different devices and internet conditions remains an ongoing effort.

Future Improvements:

➢ Knowledge Base expansion:

- Adding more plants beyond maize, coffee, avocado, and bananas.
- Including regional variations in disease patterns and treatments.

➢ Symptom Detection Logic Enhancement

- Refining the inference engine to handle multiple possible diseases and rank them by likelihood.
- Introducing a probability-based approach to suggest the most probable diagnosis instead of just one fixed result.

➢ Integrating Image Analysis for Automatic Diagnosis

- Future versions could allow users to upload pictures of diseased plants for automatic disease detection using AI-based image classification.

➢ Multilingual Support

- Adding translations to make the system usable for more farmers in different regions.

1. Team Collaboration & Report Quality

For the development of this expert system, the team was divided in 5 small groups of 1 to 3 members that were each assigned with specific tasks as follows:

Role	names	
Project Manager (Leader)	Abari Ilior Aichetou	
Data Scientist	Izere Fidella	Byiringiro Germain
AI Developer	Munyaneza Emmanuel	Denyse NYIRATUZA
Tester & Debugger	Chol Adut Gai	Abakar Sidick Baba Abakar
Technical Writer	UMUHUZA Assoumpta	GIHOZO Brigitte

Task	Deadline	Potential Risks	Solutions
Data Collection (Data Analysts)	📅 **Saturday 3 PM**	- Finding quality data might take longer than expected. - Data may need extensive cleaning.	✓ let us know if you need help ✓ AI team can suggest sources beforehand for others to help.
AI System Design & Implementation (AI developers)	📅 **Sunday 1 PM**	- Data preprocessing could take extra time. - Model training may be hard - Bugs could delay progress.	✓ **Start designing the AI model** before data arrives. ✓ **Use existing models/libraries** to speed up training. ✓ Keep **code modular** for faster debugging.
Testing Phase (Simultaneous with AI Dev)	📅 **Sunday 1 PM - Midnight**	- If AI development runs late, testing time will be reduced. - Some features may not work as expected.	✓ Start **test components early** (e.g., UI, basic functions). ✓ Have **alternative testing scenarios** prepared.
Technical Writing (Documentation)	📅 **Sunday 1 PM - Midnight**	- Delayed AI completion could slow documentation. – we might have incomplete sections.	✓ **start with general sections first** (e.g., methodology, objectives). ✓ Use a **template** to structure the report early. ✓ make sure documentation is representative of the system

Though the development process was delayed due to exam preparations, we still followed the tasks and their order. This thus allowed us to complete this work and have the working expert system deployed and available at:

https://aigroup1implementationassignment3-pvpq34r5akikb9tezwb2nr.streamlit.app/

THANK YOU